

The listing of claims will replace all prior versions, and listings, of claims in the application:

**Listing of Claims:**

1. (Currently Amended) In a JAVA computing environment, including a JAVA virtual machine, a method of generating optional attributes in a JAVA class file, said method comprising:

receiving as input JAVA runtime environment optimization information indicating JAVA application bytecodes that are associated with ~~Java~~ JAVA objects of interest of the Java runtime environment, for the particular JAVA application;

generating one or more optional attributes based on said JAVA runtime environment optimization information; and

writing said one or more optional attributes in an attribute table portion of a JAVA class file,

wherein said one or more optional attributes are processed by the JAVA virtual machine to optimize execution of the ~~Java~~ JAVA virtual machine for the particular JAVA application by controlling how the JAVA runtime environment features are provided for the objects of interest of the JAVA runtime environment are treated during execution of the particular JAVA application.

2. (Cancelled)

3. (Original) A method as recited claim 1, wherein said method further comprises:

generating computer program code that implements an application programming interface suitable for loading said one or more optional attributes.

4. (Previously Presented) A method as recited claim 3, wherein said application programming interface can be used to read said one or more optional attributes from said JAVA class file.

5. (Previously Presented) A method as recited claim 4, wherein said application programming interface includes functions that can be used to read first, last, and next optional attributes in said JAVA class file.
6. (Previously Presented) A method as recited claim 4, wherein said application programming interface includes a function suitable for finding an optional attribute in said JAVA class file.
7. (Previously Presented) A method as recited claim 1, wherein said JAVA runtime environment optimization information is stored in a database.
8. (Previously Presented) A method as recited in claim 7, wherein said database is generated by a compiler extension or a software tool suitable for analyzing a JAVA application.
9. (Original) A method as recited in claim 7, wherein said database is stored in a runtime performance manager that can interact with software modules that generate and load said one or more optional attributes.
10. (Original) A method as recited in claim 7, wherein said method further comprises:  
updating said database to reflect generation of said one or more optional attributes.
11. (Currently Amended) In a JAVA computing environment, including a JAVA virtual machine, a JAVA optional attribute generator computer-implemented method suitable for generation of optional attributes in a JAVA class file, said JAVA optional attribute generator computer-implemented method operating to:  
  
receive as input JAVA runtime environment optimization information indicating JAVA application bytecodes that are associated with ~~Java~~ JAVA objects of interest of the JAVA runtime environment, for the particular JAVA application;

generate one or more optional attributes based on said JAVA runtime environment optimization information; and

write said one or more optional attributes in an attribute table portion of a JAVA class file,

wherein said one or more optional attributes are processed by the JAVA virtual machine to optimize execution of the ~~Java~~ JAVA virtual machine for the particular JAVA application by controlling how the JAVA runtime environment features are provided for the objects of interest of the JAVA runtime environment are treated during execution of the particular JAVA application.

12. (Previously Presented) A JAVA optional attribute generator as recited in claim 11, wherein said JAVA optional attribute generator computer-implemented method operates to generate computer program code that implements an application programming interface suitable for loading said one or more optional attributes.
13. (Previously Presented) A JAVA optional attribute generator computer-implemented method as recited in claim 11, wherein an application programming interface can be used to read said one or more optional attributes from said JAVA class file.
14. (Previously Presented) A JAVA optional attribute generator computer-implemented method as recited in claim 11, wherein said JAVA runtime environment optimization information is stored in a database.
15. (Previously Presented) A JAVA optional attribute generator computer-implemented method as recited in claim 11, wherein said database is generated by a compiler extension or a software tool suitable for analyzing a JAVA application.
16. (Previously Presented) A JAVA optional attribute generator computer-implemented method as recited in claim 11,

wherein said database is stored in a runtime performance manager that can interact with software modules that generate and load said one or more optional attributes.

17. (Previously Presented) A JAVA optional attribute generator as recited in claim 11, wherein said optional attribute generator computer-implemented method operates to update said database to reflect generation of said one or more optional attributes.

18. (Previously Presented) A JAVA optional attribute generator as recited in claim 11, wherein said optional attribute generator computer-implemented method operates to generate a description of an optional attribute.

19. (Previously Presented) A JAVA optional attribute generator computer-implemented method as recited in claim 18, wherein said description is in XML format.

20. (Currently Amended) A computer readable medium including computer program code for generating optional attributes in a JAVA class file for a JAVA computing environment including a JAVA virtual machine, said computer readable medium comprising:

computer program code for receiving as input JAVA runtime environment optimization information indicating JAVA application bytecodes that are associated with ~~Java~~ JAVA objects of interest of the JAVA runtime environment, for the particular JAVA application;

computer program code for generating one or more optional attributes based on said JAVA runtime environment optimization information; and

computer program code for writing said one or more optional attributes in an attribute table portion of a JAVA class file,

wherein said one or more optional attributes are processed by the JAVA virtual machine to optimize execution of the ~~Java~~ JAVA virtual machine for the particular JAVA application by controlling how the JAVA runtime environment features are provided for the objects of interest of the JAVA runtime environment are treated during execution of the particular JAVA application.

21. (Original) A computer readable medium as recited in claim 20, wherein said method further comprises:

generating computer program code that implements an application programming interface suitable for loading said one or more optional attributes.

22. (Previously Presented) A computer readable medium as recited in claim 21, wherein said JAVA runtime environment optimization information is stored in a database.

23. (Previously Presented) A computer readable medium as recited in claim 22, wherein said database is generated by a compiler extension or a software tool suitable for analyzing a JAVA application.

24. (Original) A computer readable medium as recited in claim 22, wherein said database is stored in a runtime performance manager that can interact with software modules that generate and load said one or more optional attributes.

25. (Original) A computer readable medium as recited in claim 24, wherein said method further comprises:

updating said database to reflect generation of said one or more optional attributes.

26. (Currently Amended) The method of claim 1, wherein:

the optional attributes indicate to the JAVA virtual machine which objects of interest ~~features~~ of the ~~Java~~ JAVA runtime environment need to be loaded for the particular JAVA application.

27. (Currently Amended) The method of claim 1, wherein:

the optional attributes indicate to the JAVA virtual machine that some ~~Java~~ JAVA objects are to be given special treatment at runtime.

28. (Currently Amended) The JAVA optional attribute generator computer-implemented method of claim 11, wherein:

the optional attributes indicate to the JAVA virtual machine which objects of interest features of the ~~Java~~ JAVA runtime environment need to be loaded for the particular JAVA application.

29. (Currently Amended) The JAVA optional attribute generator computer-implemented method of claim 11, wherein:

the optional attributes indicate to the JAVA virtual machine that some ~~Java~~ JAVA objects are to be given special treatment at runtime.

30. (Currently Amended) The computer-readable medium of claim 20, wherein:

the optional attributes indicate to the JAVA virtual machine which objects of interest features of the ~~Java~~ JAVA runtime environment need to be loaded for the particular JAVA application.

31. (Currently Amended) The computer-readable medium of claim 20, wherein:

the optional attributes indicate to the JAVA virtual machine that some ~~Java~~ JAVA objects are to be given special treatment at runtime.